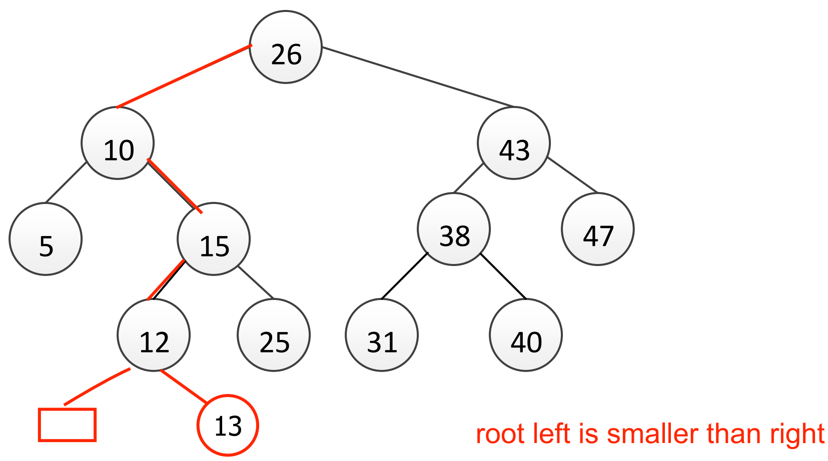
CS526 O2
Homework Assignment 5

Problem 1 (10 points). Consider the following binary search tree:



Show the resulting tree if you add the entry with key = 13 to the above tree. You need to describe, step by step, how the resulting tree is generated.

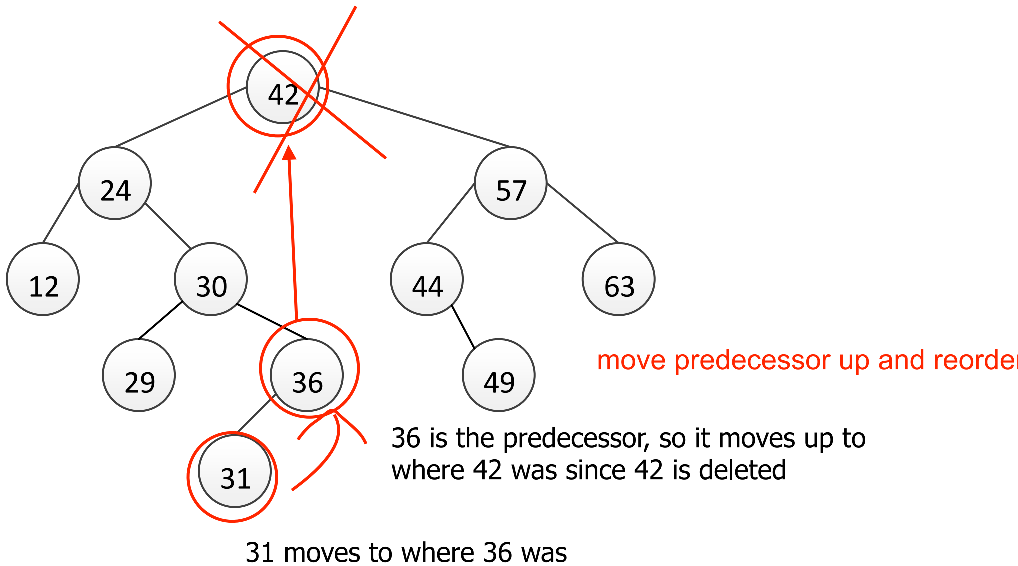
Since it is a binary search tree properties, for each internal node n with entry (k, v) pair, keys stored in the left subtree of n are less than k, right subtree of n are greater than k.

In order to do insertion to the binary search tree, we perform a search operation first.

For inserting key 13, we perform search for 13, 13 less root key 26 so we move down left to current root key 10, then we compare, since 13 greater than root key 10 so we move down to right where we get to current root key 15, and we compare 13 is less than root key 15 so we move down to now a leaf node 12, since this leaf node is not equal to the key 13, we add entry at the leaf node where the unsuccessful search ended up.

Finally, in order to keep the nature of a binary search tree, we add a placeholder to the left (the square in the graph).

Problem 2 (10 points). Consider the following binary search tree:



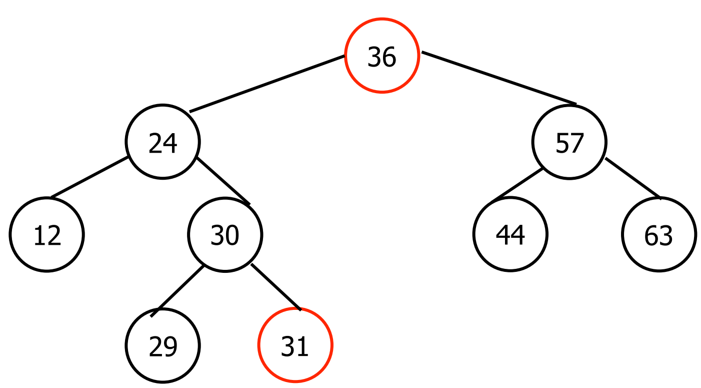
Show the resulting tree if you delete the entry with key = 42 from the above tree. You need to describe, step by step, how the resulting tree is generated.

Since it is a binary search tree properties, for each internal node n with entry (k, v) pair, keys stored in the left subtree of n are less than k, right subtree of n are greater than k.

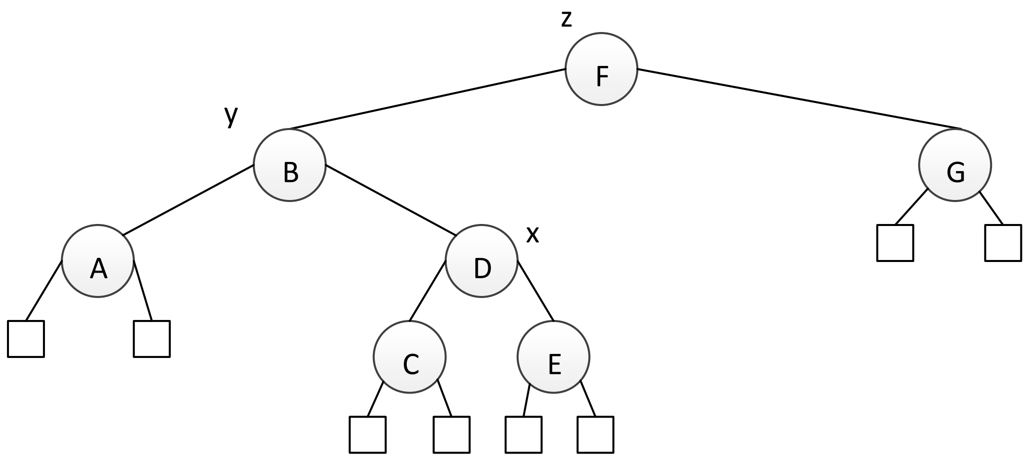
In order to do deletion to the binary search tree, we perform a search operation first. Since we found the deletion value 42 lives in the root node in the binary search tree, and the entry has two children, and both of which are internal.

First, we find the node 36 that has the largest key that is strictly less than current deletion node 42. This node is called the predecessor of 42 in the ordering of keys, which is the rightmost node in 42's left subtree. We let 36 replace 42. Since 36 is the rightmost node in 42's left subtree, it does not have a right child. It has only a left child. The node 36 is removed and the subtree rooted at 36's left child which is 31 is promoted to 36's position.

Result:



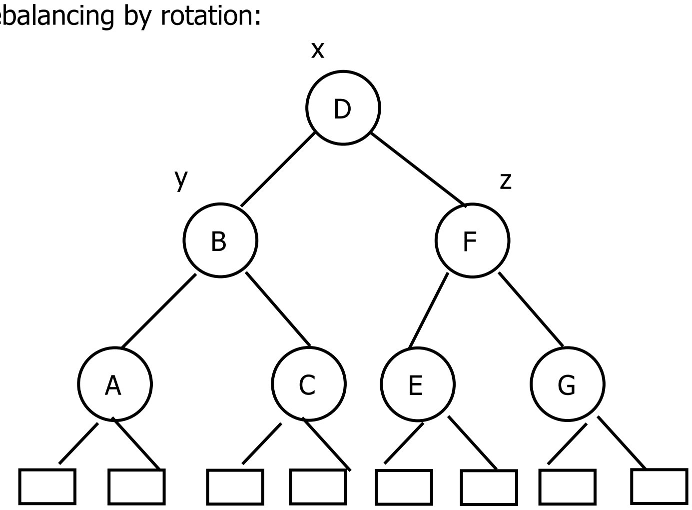
Problem 3 (10 points). Consider the following AVL tree, which is unbalanced:



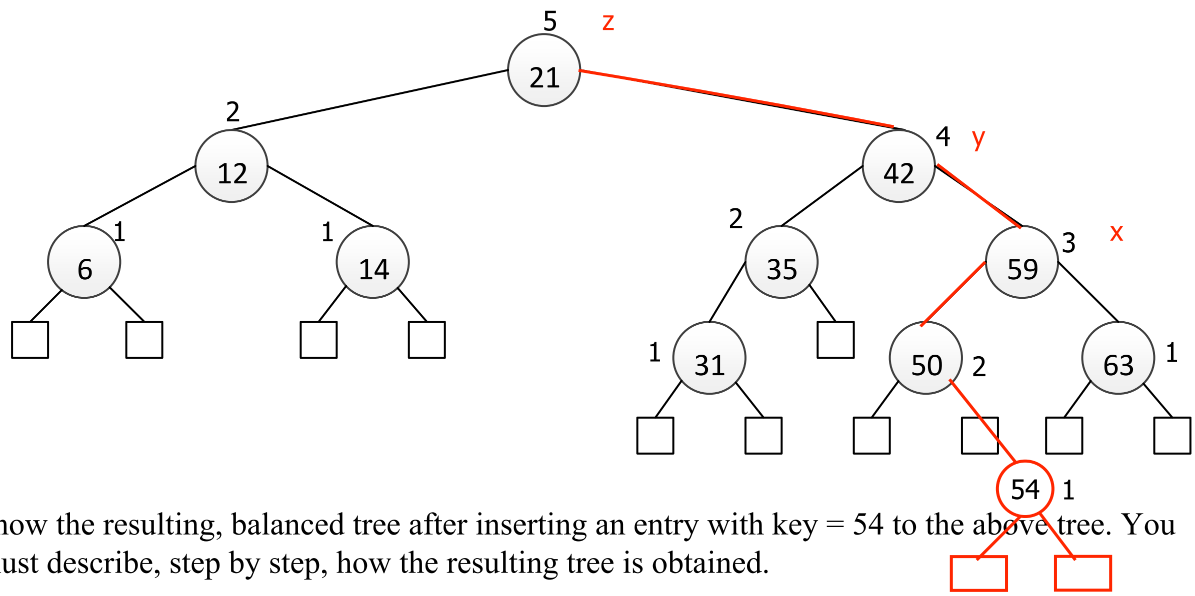
Note that the nodes *F*, *B* and *D* are labeled *z*, *y*, and *x*, respectively, following the notational convention used in the textbook. Apply a trinode restructuring on the tree and show the resulting, balanced tree.

An AVL tree is a binary search tree that satisfies the height-balance property:
For every internal node n of T, the heights of the children of n differ from at most one.
Primary operation for rebalancing the tree is rotation, and we can use search-and-repair strategy.
Search a node z that is the lowest (in height) ancestor of n that is unbalanced => F
y is z's child with the greater height => B
x is y's child with the greater height => D
Our goal here is to restructure the subtree rooted at z to reduce the path length from z to x and its subtrees.

Result After rebalancing by rotation:



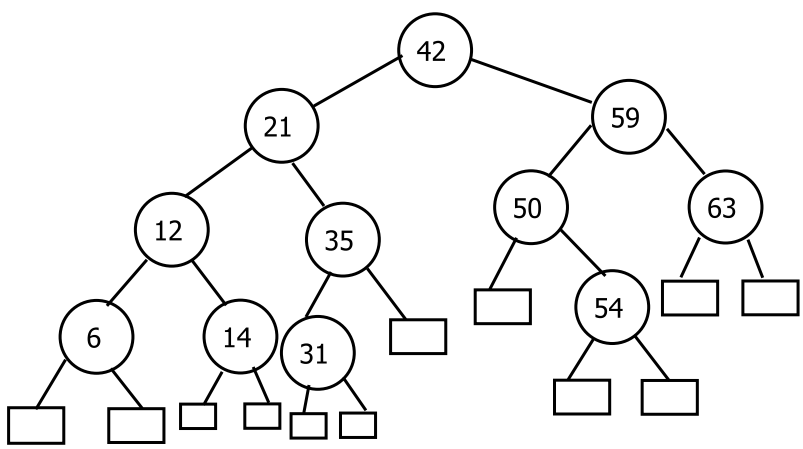
Problem 4 (10 points). Consider the following AVL tree.



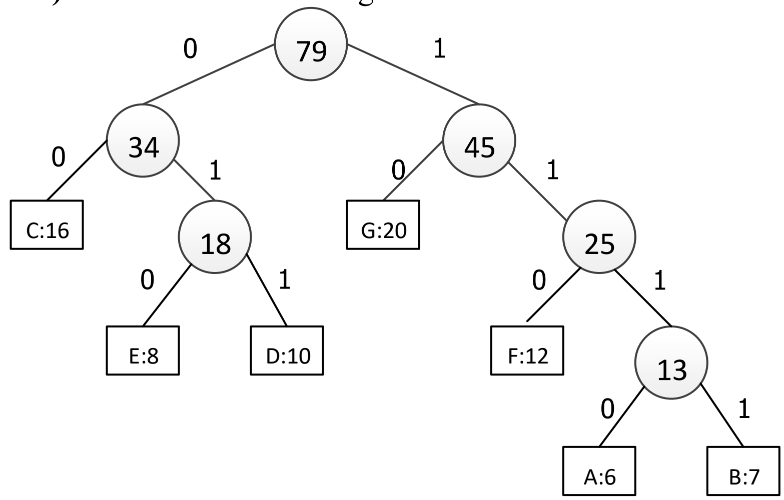
Show the resulting, balanced tree after inserting an entry with key = 54 to the above tree. You must describe, step by step, how the resulting tree is obtained.

After inserting the key 54, as path showing above, the AVL tree becomes unbalanced.
Here we are using search-and-repair strategy where we search:
a node z that is the lowest (in height) ancestor of n that is unbalanced => 21
y is z's child with the greater height => 42
x is y's child with the greater height => 59

Result After rebalancing by rotation:



Problem 5 (10 points). Consider the following Huffman tree:



(1) Encode the string "BDEGC" to a bit pattern using the Huffman tree.

(2) Decode the bit pattern "010111010011" to the original string using the Huffman tree.

Encoding "BDEGC":
B = 1111
D = 011
E = 010
G = 10
C = 00
BDEGC=11110110101000

Decode "010111010011"
010 = E
1110 = A
10 = G
011 = D
010111010011 = EAGD

Problem 6 (10 points). This question is about the *World Series* problem that we discussed in the class. The following is the probability matrix for the problem.

$P(i, j)$

							j
						1	6
					31/32	1	5
				* 26/32	15/16	1	4
			1/2	11/16	7/8	1	3
		6/32	5/16	2/1	3/4	1	2
	1/32	* 1/16	1/8	1/4	1/2	1	1
0	0	0	0	0	0		0
6	5	4	3	2	1	0	i

Calculate the probabilities of $P(4, 1)$ and $P(2, 4)$, which are marked with *. You must show all intermediate steps and calculations.

P(0,1)=1	P(1,0)=0
P(0,2)=1	P(2,0)=0
P(0,3)=1	P(3,0)=0
P(0,4)=1	P(4,0)=0
P(0,5)=1	P(5,0)=0
P(0,6)=1	P(6,0)=0

p(i, j) = 1, if i = 0 and j > 1
= 0, if i > 0 and j = 0
=(P(i-1, j) + P(i, j-1)) / 2, if i > 0 and j > 0

P(1, 1) = (P(0, 1) + P(1, 0)) / 2 = (1 + 0) / 2 = 1/2
P(2, 1) = (P(1, 1) + P(2, 0)) / 2 = (1/2 + 0) / 2 = 1/4
P(3, 1) = (P(2, 1) + P(3, 0)) / 2 = (1/4 + 0) / 2 = 1/8
P(4, 1) = (P(3, 1) + P(4, 0)) / 2 = (1/8 + 0) / 2 = 1/16 *
P(5, 1) = (P(4, 1) + P(5, 0)) / 2 = (1/16 + 0) / 2 = 1/32

P(1, 2) = (P(0, 2) + P(1, 1)) / 2 = (1 + 1/2) / 2 = 3/4
P(2, 2) = (P(1, 2) + P(2, 1)) / 2 = (3/4 + 1/4) / 2 = 1/2
P(3, 2) = (P(2, 2) + P(3, 1)) / 2 = (1/2 + 1/8) / 2 = 5/16
P(4, 2) = (P(3, 2) + P(4, 1)) / 2 = (5/16 + 1/16) / 2 = 6/32

P(1, 3) = (P(0, 3) + P(1, 2)) / 2 = (1 + 3/4) / 2 = 7/8
P(2, 3) = (P(1, 3) + P(2, 2)) / 2 = (7/8 + 1/2) / 2 = 11/16
P(3, 3) = (P(2, 3) + P(3, 2)) / 2 = (11/16 + 5/16) / 2 = 1/2

P(1, 4) = (P(0, 4) + P(1, 3)) / 2 = (1 + 7/8) / 2 = 15/16
P(2, 4) = (P(1, 4) + P(2, 3)) / 2 = (15/16 + 11/16) / 2 = 26/32 *

P(1, 5) = (P(0, 5) + P(1, 4)) / 2 = (1 + 15/16) / 2 = 31/32